8. Himanshu

Program Name: Himanshu.java **Input File: Himanshu.dat**

Himanshu's little brother loves to play the ancient game *Moksha Patam*. *Moksha Patam* originated in India in the 2nd century AD and is associated with traditional Hindu philosophy contrasting karma and kama, or destiny and desire. Today, however, the game is most known as either *Snakes and Ladders*, or *Chutes and Ladders*. Aside from its ancient roots, the game is a great tool for helping young children learn to count.

Moksha Patam is a turned based game played on a square grid with varied dimensions, in which players roll a single 6-sided die (1-6) to determine the number of moves the player is to move forward. After the end of a turn, a player may potentially land on either a "S#" signifying a snake (or chute) or an "L#" signifying a ladder. If a player lands on the lower-numbered end of a ladder, the player moves up to the ladder's higher-number square. If the player lands on the higher-numbered square of a snake, the player moves down to the snake's lower-numbered square. The player or players who reach the last square, or surpass the last square, is declared the winner. As this is a turned based game, all players are guaranteed to have the same number of turns, i.e. if Player 1 reaches or surpasses the last square, Player 2 is afforded one last move to try and reach or surpass the last square as well.

The below image shows an example *Moksha Patam* board with dimension 10 x 10:

```
99 98 S3 96 95 94 93 L3 91 90
80 81 82 83 84 85 86 87 88 89
79 S2 77 L3 75 74 73 72 71 70
60 61 62 63 64 65 66 67 68 69
59 58 57 S1 55 54 53 52 51 50
L2 41 42 43 44 45 46 S2 48 49
39 38 37 36 L1 34 33 32 31 30
20 21 22 23 24 25 26 27 28 29
L2 S1 17 16 15 14 13 12 S3 10
00 01 02 03 04 05 L1 07 08 09
```

All players start at the square with index `0`. For formatting purposes for the 10 x 10 board, `0` is denoted as `00` so that each column is straight. Assuming a player is currently at index `00` and roles a 6 from the die, the player would move to index `06`, but this index has `L1` on it denoting this as a ladder. Therefore, the player is then moved to index `35` which is the higher-number squared corresponding to `L1`. If the player were at index `91` and were to roll a 1, the player would move to index `92` and would remain here until the following turn, as this is the higher-numbered latter corresponding to `L3`. If the player was at index `54` and were to roll a 2, the player would move to index `55`. Index `55` however has `S1` denoting this as a snake (chute). Therefore, the player is moved down to index `18` as this is the lower-number square corresponding to `S1`. If a player were at index `43` and were to roll a 4, the player would move to index `47` and would remain here until the following turn, as this is the lower-numbered snake corresponding to `S2`. Although depicted in this board as being on separate rows, snakes and ladders do not necessarily have to be on the different rows. This means that a player could move horizontally only, with no vertical movement on the board.

Unfortunately, with UIL Computer Science season ramping up, Himanshu doesn't have as much time to play *Moksha Patam* with his little brother due to CS practice. In an effort to not only appease his little brother's thirst for *Moksha Patam*, but to get some extra coding practice in as well, Himansu comes up with the idea to write a program that will play *Moksha Patam* against his little brother. The game will only feature two players: Player 1 and Player 2. A series of moves for each player will be given corresponding to the dice roll for that term. As soon as a winner is found and both players have had equal number of turns, the play for the current game is terminated. Can you help Himanshu write such a program?

Input: Input starts with a line containing an integer N, the number of games to be played. N will be in range [1,20]. Each game starts first with an integer D for the dimension of the board. D will be in range [2,32]. Odd dimensions

~ Problem continues on next page ~

Himanshu, continued

for the square board are allowed. This means that the last square to be reach or surpassed is not guaranteed to be at the top left. For formatting purposes all indexes will be outputted to uniform column width, so that each column is straight up and down. The following D lines contain the board for that given game play. Boards use a back-and-forth track from the bottom of the playing area (guaranteed to be the bottom left corner) to the last square. Indexes will be incremented by 1. The board will also have scattered snakes (chutes) and ladders placed throughout the board. Snakes will be represented by `S##` and ladders will be represented by `L##`. Where `##` represents the number of the respective snake or ladder. `##` will be formatted with padded zeros to make sure the formatting for each column remains straight. For every snake or ladder present, it will be guaranteed to have a corresponding end point. It can also be guaranteed that a snake nor a ladder will be present at index `00` nor at the max index of $D^2 - 1$. Following the board are two lines. The first starting with "`P1:`" and the second starting with "`P2:`", each followed by the respected moves for the player. Moves will be in range [1,6] and separated by a comma. The number of moves present may include extra moves that won't be carried out, if a player or players, is found to have reached the max index or surpassed the max index. Last, a line of 36 dashes "`-`" is presented to separate the different game inputs.

Output: For each game played, you are to output either: `Game #CURRENT_GAME_NUMBER: Player 1 wins!`,`Game #CURRENT_GAME_NUMBER: Player 2 wins!`,`Game #CURRENT_GAME_NUMBER: Both players win!`, or `Game #CURRENT_GAME_NUMBER: Neither Player 1 or Player 2 won.` Remember, each of the two players are guaranteed to have the same number of moves, resulting in the potential for two winners.

Sample input: (the new line character is only present after all the moves for a given player have been given)

```
9
10
99 98 S3 96 95 94 93 L3 91 90
80 81 82 83 84 85 86 87 88 89
79 S2 77 L3 75 74 73 72 71 70
60 61 62 63 64 65 66 67 68 69
59 58 57 S1 55 54 53 52 51 50
L2 41 42 43 44 45 46 S2 48 49
39 38 37 36 L1 34 33 32 31 30
20 21 22 23 24 25 26 27 28 29
L2 S1 17 16 15 14 13 12 S3 10
00 01 02 03 04 05 L1 07 08 09
P1:1,4,3,1,5,6,1,5,1,3,6,1,4,5,1,6,5,1,1,2,5,2,1,4,1,3,6,3,1,3,6,1,4,5,1,6,5,1,1,2,6,5
,2,4,1,3,6,4,4,3,2,3,1,6,5,2,4,1,3
P2:6,6,1,6,2,3,3,5,1,5,1,1,1,4,3,6,4,4,3,2,3,1,6,5,2,4,1,3,6,4,4,3,2,3,1,6,5,2,4,1,3,5
,2,4,1,3,6,4,4,3,2,3,1,6,5,2,4,1,3
------------------------------------
9
99 98 S3 96 95 94 93 L3 91
80 81 82 83 84 85 86 87 88
79 S2 77 L3 75 74 73 72 71
60 61 62 63 64 65 66 67 68
59 58 57 S1 55 54 53 52 51
L2 41 42 43 44 45 46 S2 48
```

~ Sample input continues on next page ~

Himanshu, continued

```
39 38 37 36 L1 34 33 32 31
20 21 22 23 24 25 26 27 28
L2 S1 17 16 15 14 13 12 S3
P1:1,6,6,6,2
P2:1,4,3,1,5
------------------------------------


10
99 98 S3 96 95 94 93 L3 91 90
80 81 82 83 84 85 86 87 88 89
79 S2 77 L3 75 74 73 72 71 70
60 61 62 63 64 65 66 67 68 69
59 58 57 S1 55 54 53 52 51 50
L2 41 42 43 44 45 46 S2 48 49
39 38 37 36 L1 34 33 32 31 30
20 21 22 23 24 25 26 27 28 29
L2 S1 17 16 15 14 13 12 S3 10
00 01 02 03 04 05 L1 07 08 09
P1:6,6,5,6,6,6,6,6,2,5
P2:6,6,5,6,6,6,6,6,2,1
------------------------------------
10
99 98 S3 96 95 94 93 L3 91 90
80 81 82 83 84 85 86 87 88 89
79 S2 77 L3 75 74 73 72 71 70
60 61 62 63 64 65 66 67 68 69
59 58 57 S1 55 54 53 52 51 50
L2 41 42 43 44 45 46 S2 48 49
39 38 37 36 L1 34 33 32 31 30
20 21 22 23 24 25 26 27 28 29
L2 S1 17 16 15 14 13 12 S3 10
00 01 02 03 04 05 L1 07 08 09
P1:6,6,5,6,6,6,6,6,2,5
P2:6,6,5,6,6,6,6,6,2,5
------------------------------------
3
06 L1 08
05 S1 03
00 L1 S1
P1:1,1
P2:2,2
------------------------------------
```

~ Sample input continues on next page ~

Himanshu, continued

```
2
03 L1
00 L1
P1:1,1
P2:2,2
------------------------------------
5
20 L6 L5 S5 L6
L5 L4 S4 S5 S3
S3 S4 L4 L3 S4
L3 S1 L1 L2 S2
00 L1 S1 L2 S2
P1:6,6,5,6,6,6,6,6,2,5
P2:6,6,5,6,6,6,6,6,2,5
------------------------------------
11
S21 L08 S02 S16 S03 115 L06 S12 118 S10 120
L05 L17 S17 S01 L06 104 L12 S05 101 L11 S08
S15 089 L02 L07 092 S07 S09 S03 S14 L08 S11
087 S22 L12 L13 L19 082 081 S18 S20 078 077
S04 067 L04 L07 L15 071 L18 L19 S17 L17 076
L02 064 S18 062 S02 S07 L16 L14 S13 S15 S06
S14 L10 L13 S06 S16 L09 S01 051 S20 053 S12
043 L09 L11 L14 L16 S08 L03 036 L03 L18 L10
S19 L01 S10 S13 026 027 L01 029 030 031 032
S05 L04 S11 018 017 016 015 S04 S09 S21 011
000 S23 L05 003 004 005 S23 007 L15 S22 S19
P1:1,4,3,1,5,6,1,5,1,3,6,1,4,5,1,6,5,1,1,2,5,2,1,4,1,3,6,3,1,3,6,1,4,5,1,6,5,1,1,2,6,5
,2,4,1,3,6,4,4,3,2,3,1,6,5,2,4,1,3
P2:6,6,1,6,2,3,3,5,1,5,1,1,1,4,3,6,4,4,3,2,3,1,6,5,2,4,1,3,6,4,4,3,2,3,1,6,5,2,4,1,3,5
,2,4,1,3,6,4,4,3,2,3,1,6,5,2,4,1,3
------------------------------------
11
S21 L08 S02 S16 S03 115 L06 S12 118 S10 120
L05 L17 S17 S01 L06 104 L12 S05 101 L11 S08
S15 089 L02 L07 092 S07 S09 S03 S14 L08 S11
087 S22 L12 L13 L19 082 081 S18 S20 078 077
S04 067 L04 L07 L15 071 L18 L19 S17 L17 076
L02 064 S18 062 S02 S07 L16 L14 S13 S15 S06
S14 L10 L13 S06 S16 L09 S01 051 S20 053 S12
043 L09 L11 L14 L16 S08 L03 036 L03 L18 L10
S19 L01 S10 S13 026 027 L01 029 030 031 032
```

~ Sample input and output continues on next page ~

Himanshu, continued

```
S05 L04 S11 018 017 016 015 S04 S09 S21 011
000 S23 L05 003 004 005 S23 007 L15 S22 S19
P1:1
P2:6
------------------------------------
```

Sample output:

```
Game #1: Player 2 wins!
Game #2: Neither Player 1 or Player 2 won.
Game #3: Player 1 wins!
Game #4: Both players win!
Game #5: Player 1 wins!
Game #6: Both players win!
Game #7: Both players win!
Game #8: Player 2 wins!
Game #9: Neither Player 1 or Player 2 won.
```